Une condition suffisante pour l'implémentation connexionniste asynchrone

Bruno Scherrer

UNE CONDITION SUFFISANTE POUR L'IMPLÉMENTATION CONNEXIONNISTE ASYNCHRONE

Bruno Scherrer
Maia

LORIA
Campus scientifique B.P. 239
F-54506 Vandœuvre-lès-Nancy Cedex
scherrer@loria.fr

ABSTRACT
Les algorithmes connexionnistes sont inspirés de la manière dont le cerveau traite l'information : ils impliquent un grand nombre d'unités simples fortement interconnectées, manipulant des informations numériques de manière distribuée et massivement parallèle. Dans cet article, nous considérons la question de l'implémentation connexionniste asynchrone : étant donné un calcul à effectuer, est-il possible et intéressant de l'implémenter sous une forme connexionniste asynchrone ? Nous présentons une condition suffisante sur ce calcul, la contraction, pour que la réponse soit affirmative. Nous décrivons ensuite brièvement quelques problèmes intéressants qui impliquent le calcul d'une contraction, qui admettent pas conséquent une implémentation connexionniste asynchrone naturelle.

KEY WORDS
Implémentation connexionniste, asynchronisme.

Introduction

Le but de cet article est de décrire une condition suffisante, la contraction, pour qu'un calcul discret admette une solution algorithmique qui est massivement parallèle, distribuée, numérique et asynchrone, qualités de nous engloberons dans la suite par les termes "connexionniste asynchrone". Cette condition permet de *construire* facilement des réseaux connexionistes asynchrones pour des problèmes qui impliquent une contraction ; nous argumentons en particulier que cette approche est très intéressante d'un point de vue complexité temporelle. Des travaux très proches sont ceux de Lohmiller et Slotine [1] et de Wang et Slotine [2]. Dans ces articles, les auteurs étudient la stabilité de systèmes dynamiques couplés régis par des équations différentielles en utilisant des arguments de type contraction. Ils dérivent, pour de nombreuses architectures complexes, des conditions suffisantes pour que la dynamique se stabilise et illustrent ces résultats sur des problèmes de contrôle, d'observation et de synchronisation. La différence essentielle entre ces travaux et ce que nous présentons ici, est que la loi d'évolution est continue en espace tandis que (en dehors de la sous-section 2.1) nous ne faisons pas d'hypothèse de régularité sur la dynamique en espace : d'un instant au suivant, les variables d'états peuvent faire des sauts quelconques. La section 1 rappelle un résultat [3] montrant que le calcul du point fixe d'un opérateur contraction (PFOC) sur un domaine fini de taille raisonnable peut efficacement se faire à l'aide d'un algorithme connexionniste. La section 2 illustre ce résultat : nous évoquons un certain nombre de problèmes qui reposent sur le calcul de PFOC ; ces problèmes admettent des solutions connexionnistes asynchrones *naturelles*.

1 Un lien entre local et global

Considérons un réseau de processeurs élementaires fonctionnant de manière massivement parallèle et distribuée. En général, il est difficile de faire le lien entre les calculs qui sont effectués au niveau local (au niveau de chaque processeur élementaire) et au niveau global (au niveau du réseau). Il est même souvent encore plus difficile de faire ce lien lorsque les calculs locaux ne sont pas synchronisés (par une horloge centrale). Cette section rappelle un résultat de la littérature parallélisme, concernant le calcul d'un *point fixe d'un opérateur contraction* (PFOC), qui permet pour une certaine classe de calcul, de faire le lien entre local et global. Ce résultat est fondamental dans le sens où il représente une base de construction puissante pour de nombreux algorithmes connexionnsites : tout calcul qui peut être caractérisé comme le PFOC admettra une solution connexionniste asynchrone.

Considérons un ensemble fini X de taille N, une norme $\|.\|$ sur l'ensemble $\mathbb{R}^X$ des fonctions à valeurs réelles dans X, et un opérateur contraction $M : \mathbb{R}^X \longmapsto \mathbb{R}^X$, dans le sens qu'il vérifie $\|M(f') - M(f)\| \leq \gamma.\|f' - f\|$ avec $\gamma \in [0, 1)$. Le *théorème du point fixe dans les espaces de Banach* permet de dire que M a un et un seul point fixe f^*, c'est-à-dire qu'il existe une et une seule fonction f^* de $\mathbb{R}^X$ qui vérifie

$$f^* = M(f^*)$$

Une technique standard pour calculer le point fixe f^* consiste à itérer le processus suivant :

$$\forall x \in X, f^{t+1}(x) \leftarrow M(f^t)(x) \qquad (1)$$

où l'initialisation f^0 est arbitraire. On sait alors que $f^t \overset{t\to\infty}{\longrightarrow} f^*$ avec un taux de convergence linéaire γ :

$$\|f^{t+1} - f^*\| \leq \gamma.\|f^t - f^*\| \leq \gamma^{t+1}.\|f^0 - f^*\| \quad (2)$$

La complexité temporelle de ce processus itératif est le produit de deux termes :

- le temps requis pour faire une itération conformément à l'équation 1
- le nombre d'itérations pour approximer f^* avec une précision suffisante. Quelques lignes de calcul suffisent pour montrer que pour avoir une précision ϵ, il suffit d'itérer $\frac{\log\left(\frac{1}{\epsilon}\right)+\log(\|f^0-f^*\|)}{\log\left(\frac{1}{\gamma}\right)}$ fois.

Revenons au paradigme connexionniste : si X est un ensemble d'unités, si la fonction f correspond à l'activité en chaque unité, et si M est la loi de mise à jour de l'activité d'un instant à l'autre, alors la condition de contraction sur M est suffisante pour qu'on puisse déduire que l'activité du réseau connexionniste se stabilise asymptotiquement vers l'activité f^* (et ce indépendamment de l'activité initiale f_0). Par exemple, considérons un réseau récurrent (complètement connecté) de neurones formels généralisés ([4], page 4). L'évolution de l'activité de ce réseau est régie par la loi

$$f(x_i) \leftarrow \sigma(\sum_j w_{ij} f(x_j) - \mu_i) \quad (3)$$

où $f(x_i)$ est l'activité du neurone i, σ est une "fonction sigmoïde" (par exemple $x \mapsto \frac{1}{1+e^{-x}}$ ou $x \mapsto \tanh(x)$), w_{ij} est le poids de la connexions du neurone j vers le neurone i, et μ_i est le seuil du neurone i. Il est facile de voir[1] que l'activité de ce réseau récurrent vérifie la propriété de contraction pour la norme infinie si

$$\sup_x |\sigma'(x)| \max_i \sum_j |w_{ij}| < 1. \quad (4)$$

En d'autres termes, la condition 4 que les poids des connexions et la dérivée de la sigmoïde ne sont pas trop élevés est *suffisante* pour déduire que le réseau récurrent non-linéaire que nous venons de décrire converge vers une activité indépendante des valeurs initiales (mais évidemment dépendantes de w_{ij}, μ_i, σ(.), etc...).

Dans [3], les auteurs vont plus loin et montrent que le point fixe f^* peut être calculé de manière asynchrone. Si on note $M_i(f)(x)$ la $i$ème composante de $M(f)(x)$, ils montrent que le processus $f(x_i) \leftarrow M_i(f)(x_1, .., x_N)$ peut être massivement distribué sur chacun des N points $x_i \in X$ (chaque x_i peut être vu comme un processeur calculant $f(x_i)$ et fonctionnant en interaction avec l'ensemble des autres processeurs) et que f convergera toujours asymptotiquement vers f^*. Il est important de souligner que ces calculs ne nécessitent aucune synchronisation : les mises à jour de $f(x_i)$ pour tout point $x_i \in X$ peuvent se faire dans n'importe quel ordre, et même à des fréquences différentes. L'important pour la convergence asymptotique est que l'ensemble des points continuent d'être mis à jour indéfiniment. La preuve d'un tel résultat repose sur le fait qu'on peut alors écrire une variante de l'équation 2 :

$$\|f^{k_{t+1}} - f^*\| \leq \gamma.\|f^{k_t} - f^*\| \leq \gamma^{t+1}.\|f^0 - f^*\|$$

où $k_0, k_1, ...$ est une suite croissante d'instants telle que $k_0 = 0$ et chaque $f(x)$ est mis à jour au moins une fois entre les instants k_t et $k_{t+1}-1$ pour tout t (voir par exemple [5] page 27). A chaque fois que l'ensemble des valeurs $f(x)$ ont été mises à jour, on est sûr que la fonction f s'est rapprochée du point fixe f^* avec un taux linéaire γ. Du point de vue connexionniste (où X, f, et M sont respectivement un ensemble d'unités, la fonction d'activité et la loi de mise à jour), cela signifie que le réseau connexionniste, qu'il fonctionne de manière synchrone ou asynchrone, se stabilisera toujours vers le point fixe f^* de l'opérateur synchrone M.

La motivation initiale des auteurs de [3] était de montrer qu'un calcul de PFOC peut être implémenté sur des architectures parallèles ayant des délais de communications. L'équivalence entre la calcul synchrone et asynchrone montre que ces délais ne posent aucun problème. De plus, si on utilise une architecture parallèle contenant un processeur de calcul par point x (c'est souvent ce qu'on imagine quand on parle de réseau connexionniste), l'accélération induite est clairement proche de l'accélération idéale : le temps de calcul est divisé par le nombre N de processeurs. Plus de détails à propos de la complexité sur diverses architectures parallèles (synchrones et asynchrones) peuvent être trouvés dans [3].

Dans un réseau connexionniste, il est usuel d'expliciter 1) l'ensemble des *unités* et 2) sa *topologie*, c'est-à-dire l'ensemble des *connexions* entre unités. Lorsqu'on souhaite calculer le PFOC, nous avons vu qu'il est naturel d'identifier l'ensemble des unités à l'espace X. Qu'en est-il des connexions ? Dans la vision où chaque unité x_i effectue le calcul $f(x_i) \leftarrow M_i(f)(x_1, .., x_j, .., x_N)$, il faut une connexion de x_j vers x_i si la quantité $M_i(f)(x_1, .., x_N)$ dépend de x_j (c'est-à-dire si x_i a besoin de connaître x_j pour mettre à jour $f(x_i)$). Par exemple, si on regarde f comme un vecteur de dimension N et si M est une matrice de dimension $N \times N$, il faut une connexion entre les unités calculant $f(x_i)$ et $f(x_j)$ si le coefficient de M à la ligne i et à la colonne j est non nul[2]. Dans le cas le plus général, le réseau connexionniste peut être complètement connecté et récurrent : toute unité peut être connectée à toutes les autres (même à elle-même). Si, de prime abord, de tels réseaux peuvent sembler extrêmement complexes, la convergence de l'activité f vers un point fixe est assurée dès lors que M est un opérateur contraction.

Nous avons insisté sur le fait que le calcul d'un PFOC

[1] Pour tout i, $\left|\sigma(\sum_j w_{ij} f(x_j) - \mu_i) - \sigma(\sum_j w_{ij} f'(x_j) - \mu_i)\right|$
$\leq \sup_x |\sigma'(x)| \left|\sum_j w_{ij}(f(x_j) - f'(x_j))\right|$
$\leq \sup_x |\sigma'(x)| \max_i \sum_j |w_{ij}|.\|f - f'\|_\infty$.

[2] Dans le cas linéaire où M est une matrice, il est facile de voir que M est un opérateur contraction si et seulement le rayon spectral de M est strictement plus petit que 1.

est indépendant de l'initialisation f_0 de l'activitié du réseau. Ce constat a une conséquence particulièrement intéressante. Si, au cours du temps, la loi de mise à jour M évolue (comme nous venons de le voir, cela peut être lié à un changement de topologie du réseau) tout en restant une contraction, le calcul $f(x) \leftarrow M(f)(x)$ aura toujours pour effet de faire évoluer l'activité f vers le point fixe du M courant. Ceci peut être interprété comme la capacité d'une telle architecture à répondre dynamiquement à des variations de M. L'illustration d'une telle dynamicité est par exemple donnée dans [6].

En conclusion, le calcul d'un PFOC sur un domaine fini X peut être fait de manière connexionniste asynchrone ; s'il est implémenté de manière massivement parallèle, le calcul peut être significativement plus rapide (proche de l'accélération idéale) que le processus séquentiel décrit par l'équation 1. En conséquence, tout problème dont la solution peut être caractérisée comme le PFOC peut être résolu de manière accélérée par un algorithme connexionniste.

2 Illustration

Nous venons de voir que tout calcul du *point fixe d'un opérateur contraction* (PFOC) peut se faire naturellement par un algorithme connexionniste asynchrone et qu'un tel algorithme peut pleinement bénéficier de son caractère massivement parallèle. Il est alors tout de suite naturel de se demander si la classe des problèmes qui se réduisent au calcul d'un PFOC est intéressante ou pas, autrement dit si ce principe pour construire des algorithmes connexionnistes est véritablement applicable à des problèmes intéressants. Cette section a pour objectif de convaincre le lecteur que la réponse est affirmative. Nous allons évoquer quelques problèmes qui s'appuient sur des calculs de PFOC et qui admettent par conséquent une solution connexionniste asynchrone naturelle.

2.1 Recherche de zéro et minimisation sous contraintes

L'un des exemples les plus simples de contraction se rencontre lorsqu'on cherche à trouver la valeur pour laquelle une fonction s'annule. En effet, étant donné une fonction $f : \mathbb{R}^N \rightarrow \mathbb{R}^N$, considérons le problème consistant à trouver une valeur de x telle que $f(x) = 0$. Alors, une technique qui peut être utilisée pour trouver l'inconnue x consiste à effectuer l'itération $x \leftarrow x - f(x)$. L'idée sous-jacente est que si la séquence ci-dessus converge vers un certain x^*, alors ce x^* est nécessairement solution du système $f(x^*) = 0$. Comme nous l'avons vu à la partie précédente, une condition suffisante pour que cette séquence converge est que la fonction $\Phi : x \mapsto x - f(x)$ soit une contraction. Si la fonction f est dérivable, une condition qui implique la contraction est simplement que la norme du gradient de Φ soit bornée par une constante $\gamma < 1$ (voir par exemple [7]).

Le problème consistant à trouver le zéro d'une fonction peut lui-même être un sous-problème de l'optimisation non-linéaire avec contraintes d'égalités. D'une manière générale, un tel problème a la forme suivante :

$$\begin{cases} \text{minimiser} & f(x) \\ \text{avec les contraintes} & h_i(x) = 0; i = 1, 2, ..., p, \end{cases}$$

où f et h sont des fonctions régulières de $\mathbb{R}^N$ et l'une au moins de ces deux fonctions est non-linéaire. Une technique standard consiste alors à augmenter l'espace dans lequel on recherche un x qui satisfait le problème ci-dessus par autant de variables auxiliaires qu'il y a de contraintes et d'introduire une fonction lagrangienne $l(x, \lambda, \mu) = f(x) + \mu^t h(x)$ où ici nous utilisons une notation vectorielle (x est un vecteur de dimension n, μ de dimension p, etc...). La théorie de l'optimisation non-linéaire permet de transformer le problème ci-dessus en un système dans lequel on recherche un zéro ; on montre en effet [7] qu'une condition nécessaire pour qu'un point x^* soit minimum local de f est qu'il soit solution de

$$\begin{cases} \nabla_x l(x, \mu) = 0 \\ \nabla_\mu l(x, \mu) = 0. \end{cases}$$

Par conséquent la résolution d'un tel système, qui se réduit au problème de recherche de zéro que nous avons introduit au début de cette sous-section, fournit des candidats potentiels (il s'agit d'une condition nécessaire) pour être solution du problème d'opitimisation non-linéaire. Des conditions un peu plus complexes fournissent, à l'aide d'équations du même type ($F(x) = 0$), des conditions suffisantes pour que x^* soit point fixe. Nous renvoyons le lecteur à [7] pour plus de détails.

Sous des conditions favorables, les problèmes de recherche de zéro et d'optimisation linéaire (avec éventuellement des contraintes d'égalité) peuvent se ramener au calcul d'un point fixe et admettent, comme nous l'avons expliqué dans la section précédente, une résolution sous la forme d'un réseau connexionniste asynchrone. Même si les conditions pour que le processus itératif soit une contraction peut en pratique être difficile à assurer, la généralité du problème d'optimisation non-linéaire convaincra aisément le lecteur qu'il s'applique à de nombreux problèmes réels, en particulier à des "tâches cognitives". Dit autrement, le réseau connexionniste introduit dans la section précédente est un candidat naturel connexionniste pour la résolution de ces problèmes.

2.2 Contrôle optimal, apprentissage par renforcement

Un cadre qui aborde directement une tâche cognitive est celui du contrôle optimal. On considère un système contrôlé et le but est de trouver le contrôle qui optimise une certaine mesure de performance au cours de son évolution. En temps discret, le modèle central de cette approche est un processus de décision markovien (PDM) $\langle S, A, T, R \rangle$ où

- S est l'espace des états dans lequel peut se trouver le système,

- A est l'ensemble des actions (ou contrôles) que peut faire le système en chaque état,
- $T : S \times A \times S \mapsto [0; 1]$ correspond aux lois de la dynamique du système en fonction des actions : $T(s, a, s')$ est la probabilité que le système arrive dans l'état s' sachant qu'il effectue l'action a dans l'état s,
- $R : S \times A \mapsto \mathbb{R}$ est une mesure de performance instantanée qui est d'autant plus élevée qu'il est "bien" de faire l'action a dans l'état s.

Une fois qu'un problème est modélisé dans ce cadre, on cherche la loi de contrôle qui optimise la performance le long des trajectoires, par exemple en optimisant un critère qui tient compte de l'ensemble des performances instantanées comme $E_{(s_t)}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t)\right]$ où s_t est la variable aléatoire induite par une loi de contrôle (s_t est aléatoire donc on prend l'espérance) et γ est un facteur d'actualisation qui assure que le critère est fini. Dans ce cas, on montre qu'il suffit de chercher des lois de contrôle déterministes et sans mémoire, c'est-à-dire de la forme $\pi : S \mapsto A$ et qu'une loi de contrôle π^* qui optimise (globalement et pour tout état intial s_0) le critère que nous venons d'introduire vérifie :

$$\pi^*(s) = \arg\max_{a \in A} R(s, a) + \gamma \sum_{s'} T(s, a, s') V^*(s')$$

où $V^* : S \mapsto \mathbb{R}$ est une fonction qui vérifie une équation du type point fixe :

$$V^*(s) = \max_{a \in A} R(s, a) + \gamma \sum_{s'} T(s, a, s') V^*(s').$$

Récrivons l'équation ci-dessus $V^* = BV^*$; on peut vérifier que l'opérateur B est un opérateur de contraction (de facteur de contraction au maximum γ). Autrement dit, et conformément à la première section, il est immédiat de construire un réseau connexionniste asynchrone qui résout le problème du contrôle optimal : dans ce réseau, on identifie unités et états du PDM, et il faut une connexion entre 2 états s et s' si la transition $s \to s'$ est possible.

Nous venons de décrire la version à temps discret du contrôle optimal. Il existe également une version à temps continu. La fonction décrivant la dynamique T est alors remplacée par une équation différentielle stochastique et on montre que l'équivalent de la fonction V^* ci-dessus vérifie une équation aux dérivées partielles (EDP), du type *Hamilton-Jacobi-Bellman*. En pratique, une méthode pour calculer une solution approchée de cette EDP consiste à construire un schéma de discrétisation (aux différences finies ou aux éléments finis), dont on peut montrer (pour un niveau de discrétisation fixé) qu'il est équivalent à un certain problème à temps discret, c'est-à-dire à un MDP qu'on peut résoudre de manière connexionniste et asynchrone.

Une variante du problème du contrôle optimal (à temps discret comme à temps continu) dans laquelle les paramètres T et R décrivant la dynamique et la performance immediate sont initialement inconnus mais peuvent être estimés statistiquement à l'aide d'expériences/simulations, s'appelle l'apprentissage par renforcement. Ce problème constitue un modèle informatique simple et non trivial d'un agent apprenant tout en interagissant avec un monde dans lequel il reçoit des punitions et des récompenses. En se basant sur les implémentations connexionnistes évoquées plus haut, et en rajoutant des estimations statistiques standards aux éléments du réseau, on peut facilement construire une architecture connexionniste asynchrone qui résout le problème de l'apprentissage par renforcement. Ceci est décrit de manière plus détaillée dans [8] : on y montre que l'estimation statistique de la loi de la dynamique T prend la forme d'une loi hebbienne sur les connexions de l'architecture et on propose des solutions pour le cas où l'espace d'états est très grand.

Conclusion

Dans cet article, nous avons montré que des calculs de type point fixe d'un opérateur contraction (PFOC) se prêtaient particulièrement bien à une implémentation connexionniste asynchrone. Nous avons de plus illustré cette idée en présentant divers domaines et problèmes où les objets mathématiques qu'il s'agit de calculer sont des PFOC : recherche de zéro, optimisation non-linéaire, contrôle optimal (à temps discret et continu), apprentissage par renforcement. Ces problèmes ont donc des solutions connexionnistes asynchrones naturelles.

Références

[1] W. Lohmiller and J.J.E. Slotine. Contraction analysis for nonlinear systems. *Automatica*, 34(6), 1998.

[2] W. Wang and J.J.E. Slotine. On partial contraction analysis for coupled nonlinear oscillators. *Biological Cybernetics*, 92(1), 2005.

[3] D.P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation : Numerical Methods*. Prentice hall, 1989.

[4] J. Hertz, A. Krogh, and R. Palmer. *Introduction to the theory of neural computation*. Addison-Wesley, Reedwood City, 1991.

[5] D.P. Bertsekas and J.N. Tsitsiklis. *Neurodynamic Programming*. Athena Scientific, 1996.

[6] B. Scherrer. *Apprentissage de représentation et auto-organisation modulaire pour un agent autonome*. PhD thesis, Université Henri Poincaré - Nancy 1, January 2003.

[7] D. Luenberger. *Linear and nonlinear programming*. Addison-Wesley, New York, 1989.

[8] B. Scherrer. Neurocomputing for optimal control and reinforcement learning with large state space. *Neurocomputing*, 63 :229–251, 2005.